Flexible Models for Secure Systems

Sarah Meiklejohn
A creation story

Motivating scenario that founded modern cryptography:
A creation story

Motivating scenario that founded modern cryptography:
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob!
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob! (CC#) (Browser) → (Web server)
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob! (CC#)

(Browser) -> (Web server)

Security model for this interaction is well established
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob! (CC#)

(Browser) ➔ [?] ➔ (Web server)

Security model for this interaction is well established
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob!

(Browser) → ? → (Web server)

Security model for this interaction is well established

Encryption works for secure online communication: SSL/TLS [1996]
A creation story

Motivating scenario that founded modern cryptography:

Hi Bob!  
(BC#)  
(Browser)  
encryption  
(Web server)

Security model for this interaction is well established

Encryption works for secure online communication: SSL/TLS [1996]
Fast-forward 15 years...
Fast-forward 15 years…
Fast-forward 15 years…
Fast-forward 15 years…
Fast-forward 15 years…
Fast-forward 15 years…
Fast-forward 15 years…
Fast-forward 15 years…

**Problem #1:** everyone is an adversary!
Fast-forward 15 years…

Problem #1: **everyone is an adversary!**
Fast-forward 15 years…

Problem #1: everyone is an adversary!
Fast-forward 15 years…

Problem #1: everyone is an adversary!
Fast-forward 15 years…

Problem #1: everyone is an adversary!
Fast-forward 15 years…

Problem #2: what are the security goals?
Fast-forward 15 years…

Problem #2: what are the security goals?
Fast-forward 15 years…

Problem #2: **what are the security goals?**

- can Bob retrieve Alice’s files?
Fast-forward 15 years…

Problem #2: what are the security goals?

• can Bob retrieve Alice’s files?
• can Dropbox read Alice’s files?
Fast-forward 15 years…

Problem #2: what are the security goals?

- can Bob retrieve Alice’s files?
- can Dropbox read Alice’s files?
- did Dropbox delete Alice’s files?
Fast-forward 15 years…

Problem #2: **what are the security goals?**

- can Bob retrieve Alice’s files?
- can Dropbox read Alice’s files?
- did Dropbox delete Alice’s files?
- can Dropbox efficiently store files?
Fast-forward 15 years…

Problem #2: what are the security goals?

• can Bob retrieve Alice’s files?
• can Dropbox read Alice’s files?
• did Dropbox delete Alice’s files?
• can Dropbox efficiently store files?
• ?
A spectrum of solutions
A spectrum of solutions
A spectrum of solutions

theory

practice
A spectrum of solutions

e-cash [C82]
A spectrum of solutions

e-cash [C82]
A spectrum of solutions

outsourced comp. [GGP10]
e-cash [C82]
A spectrum of solutions

RKA [B93, K93]
MPC [Y82]
FHE [G09]
outsourced comp. [GGP10]
e-cash [C82]
ORAM [GO96]
A spectrum of solutions

theory

RKA [B93,K93]
MPC [Y82]
FHE [G09]
outsourced comp. [GGP10]
e-cash [C82]
ORAM [GO96]

practice
A spectrum of solutions

- RKA [B93, K93]
- MPC [Y82]
- FHE [G09]
- outsourced comp. [GGP10]
- e-cash [C82]
- ORAM [GO96]

Wickr

theory | practice
A spectrum of solutions

theory

RKA [B93,K93]
MPC [Y82]
e-cash [C82]
outsourced comp. [GGP10]

practice

FHE [G09]
ORAM [GO96]

Wickr
A spectrum of solutions

theory

- RKA [B93, K93]
- MPC [Y82]
- outsourced comp. [GGP10]
- e-cash [C82]

practice

- Wickr
- diaspora*
- ORAM [GO96]
A spectrum of solutions

- RKA [B93,K93]
- FHE [G09]
- MPC [Y82]
- outsourced comp. [GGP10]
- e-cash [C82]
- ORAM [GO96]
- Bitcoin
- TrueCrypt
- Tor
- diaspora*
- SSL/TLS
- Wickr
- PGP

theory

practice
A spectrum of solutions

<table>
<thead>
<tr>
<th>Theory</th>
<th>Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>RKA [B93,K93]</td>
<td>SSL/TLS</td>
</tr>
<tr>
<td>MPC [Y82]</td>
<td>Wickr</td>
</tr>
<tr>
<td>FHE [G09]</td>
<td>TrueCrypt</td>
</tr>
<tr>
<td>outsourced comp.</td>
<td>Tor</td>
</tr>
<tr>
<td>[GGP10]</td>
<td>diaspora*</td>
</tr>
<tr>
<td>e-cash [C82]</td>
<td>PGP</td>
</tr>
<tr>
<td>ORAM [GO96]</td>
<td></td>
</tr>
</tbody>
</table>

theory

practice
A spectrum of solutions

<table>
<thead>
<tr>
<th>Theory</th>
<th>Practice</th>
</tr>
</thead>
<tbody>
<tr>
<td>RKA [B93,K93]</td>
<td>SSL/TLS</td>
</tr>
<tr>
<td>MPC [Y82]</td>
<td>TrueCrypt</td>
</tr>
<tr>
<td>FHE [G09]</td>
<td>Wickr</td>
</tr>
<tr>
<td>outsourced comp. [GGP10]</td>
<td>diasporea*</td>
</tr>
<tr>
<td>e-cash [C82]</td>
<td>PGP</td>
</tr>
<tr>
<td>ORAM [GO96]</td>
<td></td>
</tr>
</tbody>
</table>

theory

practice
My research

[MSF10, LM13]  [BMT14]  [MEKHL10]  [MMCS11]  [MMS11]

[CM14]  [CKLM12, 13a, 13b]  [CMZ13]  [MP+13, HDM+14]  [OMSK13]
My research

<table>
<thead>
<tr>
<th>Reference</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>[MSF10,LM13]</td>
<td><img src="MSF10,LM13.png" alt="Image" /></td>
</tr>
<tr>
<td>[BMT14]</td>
<td><img src="BMT14.png" alt="Image" /></td>
</tr>
<tr>
<td>[MEKHL10]</td>
<td><img src="MEKHL10.png" alt="Image" /></td>
</tr>
<tr>
<td>[MMCS11]</td>
<td><img src="MMCS11.png" alt="Image" /></td>
</tr>
<tr>
<td>[MMS11]</td>
<td><img src="MMS11.png" alt="Image" /></td>
</tr>
<tr>
<td>[CM14]</td>
<td><img src="CM14.png" alt="Image" /></td>
</tr>
<tr>
<td>[CKLM12,13a,13b]</td>
<td><img src="CKLM12,13a,13b.png" alt="Image" /></td>
</tr>
<tr>
<td>[CMZ13]</td>
<td><img src="CMZ13.png" alt="Image" /></td>
</tr>
<tr>
<td>[MP+13,HDM+14]</td>
<td><img src="MP+13,HDM+14.png" alt="Image" /></td>
</tr>
<tr>
<td>[OMSK13]</td>
<td><img src="OMSK13.png" alt="Image" /></td>
</tr>
</tbody>
</table>

- RKA
- Bitcoin
My research

[MSF10, LM13] [BMT14] [MEKHL10] [MMCS11] [MMS11]

[CM14] [CKLM12,13a,13b] [CMZ13] [MP+13,HDM+14] [OMSK13]

RKA

Bitcoin
My research

<table>
<thead>
<tr>
<th>Reference</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[MSF10,LM13]</td>
<td></td>
</tr>
<tr>
<td>[BMT14]</td>
<td></td>
</tr>
<tr>
<td>[MEKHL10]</td>
<td></td>
</tr>
<tr>
<td>[MMCS11]</td>
<td></td>
</tr>
<tr>
<td>[MMS11]</td>
<td></td>
</tr>
<tr>
<td>[CM14]</td>
<td></td>
</tr>
<tr>
<td>[CKLM12,13a,13b]</td>
<td></td>
</tr>
<tr>
<td>[CMZ13]</td>
<td></td>
</tr>
<tr>
<td>[MP+13,HDM+14]</td>
<td></td>
</tr>
<tr>
<td>[OMSK13]</td>
<td></td>
</tr>
</tbody>
</table>

RKA

Bitcoin
My research

- [MSF10, LM13]
- [BMT14]
- [MEKHL10]
- [MMCS11]
- [MMS11]
- [CM14]
- [CKLM12, 13a, 13b]
- [CMZ13]
- [MP+13, HDM+14]
- [OMSK13]

RKA

Bitcoin
Digital signatures are everywhere
Digital signatures are everywhere
Digital signatures are everywhere

<table>
<thead>
<tr>
<th>Address</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>17ijmE4nZstNnevHlawnhCW9nZ</td>
<td>0.37 BTC</td>
</tr>
<tr>
<td>1cTHK2B4D3qq4kj44wxfeQCTE6N</td>
<td>3.11 BTC</td>
</tr>
<tr>
<td>1ZhKkzRLfpNGNHf8qDDH7uRZta</td>
<td>0.20 BTC</td>
</tr>
<tr>
<td>1QJ3FK8grGXKEPqVUC1tx6G5Hw</td>
<td>5.16 BTC</td>
</tr>
</tbody>
</table>

The server’s host key is not cached in the registry. You have no guarantee that the server is the computer you think it is. The server’s rsa2 key fingerprint is: ssh-rsa 1024 0a:27:d5:4f:00:9c:d1:a3:ff:ad:5c:cd:b3:7c:83:42 If you trust this host, hit Yes to add the key to PuTTY’s cache and carry on connecting. If you want to carry on connecting just once, without adding the key to the cache, hit No. If you do not trust this host, hit Cancel to abandon the connection.
Digital signatures are everywhere
Digital signatures are everywhere
Practical security of digital signatures
Practical security of digital signatures

Signatures are proved secure in standard cryptographic models

\[ \text{Sign}(sk, m) \quad \text{Verify}(pk, m, \sigma) \]

\[ \text{KeyGen} \]

\[ m \quad \sigma \]
Practical security of digital signatures

Signatures are proved secure in standard cryptographic models

Standard model is useless against side channels and fault injection
Practical security of digital signatures

Signatures are proved secure in **standard cryptographic models**

Standard model is useless against side channels and fault injection
Practical security of digital signatures

Signatures are proved secure in **standard cryptographic models**

Standard model is useless against side channels and fault injection
Practical security of digital signatures

Signatures are proved secure in **standard cryptographic models**

Standard model is useless against side channels and fault injection
Practical security of digital signatures

Signatures are proved secure in standard cryptographic models

Standard model is useless against side channels and fault injection

RKA (related key attack) security considers these attacks
Practical security of digital signatures

Signatures are proved secure in **standard cryptographic models**

```
Sign(sk, m) → σ
Verify(pk, m, σ)
```

**Standard model is useless** against side channels and fault injection

**RKA (related key attack) security** considers these attacks

```
Sign(φ(sk), m) → m′, σ′
```

Our research can help **create better RKA schemes**
Standard definitions for signatures [GMR88]
Standard definitions for signatures [GMR88]
Standard definitions for signatures [GMR88]
Standard definitions for signatures [GMR88]

\[ \text{Sign}(sk,m) \]

\[ \text{KeyGen} \rightarrow pk \]

\[ \text{sk} \]

\[ m \]

\[ \sigma \]
Standard definitions for signatures [GMR88]

\[ \text{Sign}(sk,m) \]

\[ \sigma \]

\[ \text{Verify}(pk,m,\sigma) \]
Standard definitions for signatures [GMR88]

\[ \text{Sign}(sk, m) \]

\[ \text{Verify}(pk, m, \sigma) \]

\[ \text{KeyGen} \]

\[ m \]

\[ \sigma \]

\[ sk \]

\[ pk \]
Standard definitions for signatures [GMR88]

Sign(sk,m) → sk

KeyGen → pk

σ

Verify(pk,m,σ)

m

m

pk
Standard definitions for signatures [GMR88]
Standard definitions for signatures [GMR88]

A wins if (1) $\text{Verify}(pk, m, \sigma') = 1$ and (2) it didn’t query $m'$ to oracle
Standard definitions for signatures [GMR88]

\[ \text{Sign}(sk,m) \xrightarrow{\sigma} \text{Verify}(pk,m,\sigma) \]

\[ \text{KeyGen} \xrightarrow{pk} \]

A wins if (1) \( \text{Verify}(pk,m',\sigma') = 1 \) and (2) it didn’t query \( m' \) to oracle
Standard definitions for signatures [GMR88]

\[ \text{Sign}(sk, m) \quad \text{KeyGen} \quad \text{pk} \]

\[ \text{Verify}(pk, m, \sigma) \]

A wins if (1) \( \text{Verify}(pk, m', \sigma') = 1 \) and (2) it didn’t query \( m' \) to oracle.

**Problem:** This assumption is violated by tampering [AK96,…], side channels [W91,KJJ99,…] and fault injection [BS97,BdML97,…]
Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)
Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors $N$ given one faulty signature (attack also applies to Rabin signatures, and general RSA)

$$\sigma = H(m)^d \mod N$$
Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)

\[ \sigma = H(m)^d \mod N \]
Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors $N$ given one faulty signature (attack also applies to Rabin signatures, and general RSA)

$$\sigma = H(m)^d \mod N$$

any perturbation at all!
Related key attacks (RKA)

Attack on RSA-CRT [BdML97,L97] factors N given one faulty signature (attack also applies to Rabin signatures, and general RSA)

\[ \sigma = H(m)^d \mod N \]

Any perturbation at all!
φ-RKA-secure signatures [BCM11]
\( \phi \)-RKA-secure signatures [BCM11]
\( \phi \)-RKA-secure signatures [BCM11]
φ-RKA-secure signatures [BCM11]
\(\phi\)-RKA-secure signatures [BCM11]

A wins if (1) \(\text{Verify}(pk, m', \sigma') = 1\) and (2) it didn’t get \(\sigma'\) from oracle (when querying on \(\phi = \text{id}\))
φ-RKA-secure signatures [BCM11]

A wins if
1. \( \text{Verify}(pk, m', \sigma') = 1 \)
2. it didn’t get \( \sigma' \) from oracle (when querying on \( \phi = id \))

\( \Phi = \{id\} \) gives standard unforgeability
φ-RKA-secure signatures [BCM11]

A wins if (1) $\text{Verify}(pk, m', \sigma') = 1$ and (2) it didn’t get $\sigma'$ from oracle (when querying on $\phi=id$)

$\Phi=\{id\}$ gives standard unforgeability

$\Phi=\{all\ functions\}$ isn’t possible [BK03]
φ-RKA-secure signatures [BCM11]

A wins if (1) $\text{Verify}(pk, m', \sigma') = 1$ and (2) it didn’t get $\sigma'$ from oracle (when querying on $\phi = \text{id}$)
\( \phi \)-RKA-secure signatures [BCM11]

A wins if (1) \( \text{Verify}(pk, m', \sigma') = 1 \) and (2) it didn’t get \( \sigma' \) from oracle (when querying on \( \phi = \text{id} \))

[BdML97] shows that RSA-CRT is not \( \phi \)-RKA-secure for non-trivial \( \phi \)
**φ-RKA-secure signatures [BCM11]**

A wins if (1) $\text{Verify}(pk, m', \sigma') = 1$ and (2) it didn’t get $\sigma'$ from oracle (when querying on $\phi = \text{id}$)

[BdML97] shows that RSA-CRT is **not φ-RKA-secure** for non-trivial $\phi$

**Problem:** $\phi$-RKA schemes are really hard to construct (for interesting classes $\phi$; for most primitives)
Our construction [Bellare M Thomson Eurocrypt14]

ϕ-RKA-OWF
Our construction [Bellare M Thomson Eurocrypt14]

\( \phi \text{-RKA-OWF} \)
Our construction [Bellare M Thomson Eurocrypt14]
Our construction [Bellare M Thomson Eurocrypt14]

\[ \phi\text{-RKA-OWF} \quad \rightarrow \quad \phi\text{-RKA signature} \]

\[ \text{SE-NIZK} \]
Our construction [Bellare M Thomson Eurocrypt14]

simple construction [DHLW10, CKLM14]

- **KeyGen**: $\text{crs} \leftarrow \text{CRSGen}; \ x \leftarrow \text{Dom}(f); \ y \leftarrow f(x)$
  
  return (pk=(crs,y),sk=x)

- **Sign**(sk,m): return Prove(crs,y||m,x)

- **Verify**(pk,σ,m): return Verify(crs,y||m,σ)
Our construction [Bellare \textbf{M} Thomson Eurocrypt14]

\begin{itemize}
  \item \textbf{KeyGen}: \text{crs} \leftarrow \text{CRSGen}; \text{x} \leftarrow \text{Dom}(f); \text{y} \leftarrow f(\text{x})
    \text{return (pk=(crs,y),sk=x)}
  \item \textbf{Sign}(sk,m): \text{return Prove(crs,y||m,x)}
  \item \textbf{Verify}(pk,\sigma,m): \text{return Verify(crs,y||m,}\sigma)\text{ proof of knowledge of secret key}
\end{itemize}
Our construction [Bellare M Thomson Eurocrypt14]

Joint Enc/Sig

KDM storage

φ-RKA-OWF

SE-NIZK

φ-RKA signature

simple construction [DHLW10,CKLM14]

• **KeyGen**: \( \text{crs} \leftarrow \text{CRSGen}; \ x \leftarrow \text{Dom}(f); \ y \leftarrow f(x) \)
  
  return \((\text{pk}=(\text{crs},y),\text{sk}=x)\)

• **Sign**(sk,m): return Prove(\(\text{crs},y\Vert m,x\))

• **Verify**(pk,σ,m): return Verify(\(\text{crs},y\Vert m,\sigma\))

proof of knowledge of secret key
Our construction [Bellare M Thomson Eurocrypt14]

Signature inherits $\Phi$-RKA security from one-way function
Our construction [Bellare M Thomson Eurocrypt14]

Signature inherits $\Phi$-RKA security from one-way function
Our construction [Bellare M Thomson Eurocrypt14]

Signature inherits $\Phi$-RKA security from one-way function

Many natural one-way functions (e.g., RSA) are $\Phi$-RKA-secure with no additional assumptions
Our construction [Bellare M Thomson Eurocrypt14]

Signature inherits $\Phi$-RKA security from one-way function

Many natural one-way functions (e.g., RSA) are $\Phi$-RKA-secure with no additional assumptions

Creating new RKA-secure signatures is easier!
Takeaway

Problem: $\phi$-RKA schemes are really hard to construct (for interesting classes $\phi$; for most primitives)
φ-RKA-secure one-way functions are natural

Problem: φ-RKA schemes are really hard to construct (for interesting classes φ; for most primitives)
Takeaway

Problem: $\phi$-RKA schemes are really hard to construct (for interesting classes $\phi$; for most primitives)

$\phi$-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
Takeaway

Problem: $\phi$-RKA schemes are really hard to construct
(for interesting classes $\phi$; for most primitives)

$\phi$-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x)=g^x$) is secure w.r.t. linear functions
Takeaway

Problem: $\phi$-RKA schemes are really hard to construct (for interesting classes $\phi$; for most primitives)

$\phi$-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x) = g^x$) is secure w.r.t. linear functions
- Learning with errors ($f(s,e) = As + e \mod q$) is secure w.r.t. addition
Takeaway

Problem: $\phi$-RKA schemes are really hard to construct (for interesting classes $\phi$; for most primitives)

$\phi$-RKA-secure one-way functions are natural

- RSA function is secure w.r.t. exponentiation
- Exponentiation ($f(x) = g^x$) is secure w.r.t. linear functions
- Learning with errors ($f(s,e) = As + e \mod q$) is secure w.r.t. addition

Our result can pave the way for easier RKA constructions
My research

- [MSF10, LM13]
- [BMT14]
- [MEKHL10]
- [MMCS11]
- [MMS11]
- [CM14]
- [CKLM12,13a,13b]
- [CMZ13]
- [MP+13, HDM+14]
- [OMSK13]

- RKA
- Bitcoin
My research

References:

[MSF10, LM13]  [BMT14]  [MEKHL10]  [MMCS11]  [MMS11]

[CM14]  [CKLM12, 13a, 13b]  [CMZ13]  [MP+13, HDM+14]  [OMSK13]

RKA  Bitcoin
What is Bitcoin?
What is Bitcoin?
What is Bitcoin?

Centralized
What is Bitcoin?

Centralized

Decentralized
What is Bitcoin?

Centralized
Real-world identities

Decentralized
What is Bitcoin?

Centralized
Real-world identities

Decentralized
Pseudonyms
What is Bitcoin?

Centralized
Real-world identities
Non-public transactions

Decentralized
Pseudonyms
What is Bitcoin?

Centralized
Real-world identities
Non-public transactions

Decentralized
Pseudonyms
Public transactions
What is Bitcoin?

Centralized
Real-world identities
Non-public transactions
Regulated

Decentralized
Pseudonyms
Public transactions
What is Bitcoin?

**Centralized**
- Real-world identities
- Non-public transactions
- Regulated

**Decentralized**
- Pseudonyms
- Public transactions
- Unregulated*
What is Bitcoin?

**Centralized**
- Real-world identities
- Non-public transactions
- Regulated
- Not anonymous

**Decentralized**
- Pseudonyms
- Public transactions
- Unregulated*
What is Bitcoin?

Centralized
Real-world identities
Non-public transactions
Regulated
Not anonymous

Decentralized
Pseudonyms
Public transactions
Unregulated*
Potentially anonymous
Why study Bitcoin?
Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity
Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present. Distinct Challenges for Deterring Illicit Activity.

Ponzi-Scheme Charge Is Good News for Bitcoin
Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Ponzi-Scheme Charge Is Good News for Bitcoin

Estimated 18 percent of US drug users bought from Silk Road, says study
Why study Bitcoin?

(U) Bitcoin Virtual Currency: Unique Features Present Distinct Challenges for Deterring Illicit Activity

Ponzi-Scheme Charge Is Good News for Bitcoin

Estimated 18 percent of US drug users bought from Silk Road, says study

How much anonymity does Bitcoin really provide?
Bitcoin’s explosive growth
Bitcoin’s explosive growth

we buy 30 bitcoins at $5/BTC
Bitcoin’s explosive growth

we buy 30 bitcoins at $5/BTC
Bitcoin’s explosive growth

Jan'11 Jan'13 Jan'14

$/BTC

we buy 30 bitcoins at $5/BTC

250
Bitcoin’s explosive growth

we buy 30 bitcoins at $5/BTC

Jan’11  Jan’13  Jan’14
Bitcoin's explosive growth

we buy 30 bitcoins at $5/BTC
Bitcoin’s explosive growth

*current market capitalization of > $10B!*

we buy 30 bitcoins at $5/BTC

Jan’11 Jan’13 Jan’14
How does Bitcoin work?
How does Bitcoin work?

- decentralized
How does Bitcoin work?

- decentralized
- transfer money
How does Bitcoin work?

- decentralized
- transfer money

pk_A

pk_B

public key address
How does Bitcoin work?

- decentralized
- transfer money

public key address

\[ pk_A \quad \rightarrow \quad 1 \quad \rightarrow \quad pk_B \]
How does Bitcoin work?

- decentralized
- transfer money

\[
m = (pk_A, pk_B, 1) \\
\sigma = \text{Sign}(sk_A, m) \\
tx = (m, \sigma)
\]
How does Bitcoin work?

- decentralized
- transfer money

pk\textsubscript{A}  
m = (pk\textsubscript{A}, pk\textsubscript{B}, 1)  
\sigma = \text{Sign}(sk\textsubscript{A}, m)  
tx = (m, \sigma)

Verify

public key address
How does Bitcoin work?

- decentralized
- transfer money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ tx = (m, \sigma) \]
How does Bitcoin work?

- decentralized
- transfer money
- generate money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ \text{tx} = (m, \sigma) \]
How does Bitcoin work?

- decentralized
- transfer money
- generate money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ tx = (m, \sigma) \]
How does Bitcoin work?

- decentralized
- transfer money
- generate money

$m = (pk_A, pk_B, 1)$

$\sigma = \text{Sign}(sk_A, m)$

$tx = (m, \sigma)$

public key address
How does Bitcoin work?

- decentralized
- transfer money
- generate money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ tx = (m, \sigma) \]
How does Bitcoin work?

- decentralized
- transfer money
- generate money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ tx = (m, \sigma) \]
How does Bitcoin work?

- decentralized
- transfer money
- generate money

\[ m = (pk_A, pk_B, 1) \]
\[ \sigma = \text{Sign}(sk_A, m) \]
\[ tx = (m, \sigma) \]
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- prevent double-spending
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- **prevent double-spending**
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- prevent double-spending

25 \rightarrow tx_1 \rightarrow 10 \rightarrow tx_2 \rightarrow 25 \rightarrow 34
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- prevent double-spending

To spend bitcoins, a user must indicate the previous transaction.
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- prevent double-spending

To spend bitcoins, a user must indicate the previous transaction.
To spend bitcoins, a user must indicate the previous transaction.

- decentralized
- transfer money
- generate money
- prevent double-spending

1. Start with tx1: 25
2. Next is tx2: 10 and 25
3. Then tx3: 1 and 34
4. Finally tx3: 2 and 32

m = (tx1, tx2, tx3)
How do bitcoins get spent?

- decentralized
- transfer money
- generate money
- prevent double-spending

To spend bitcoins, a user must indicate the previous transaction

All bitcoins received in a transaction must be spent all at once
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys.
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys.
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result, the Bitcoin graph is complicated and has 12 million public keys.

Collapse into a more manageable graph of clusters of public keys representing distinct entities.
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys.

Collapse into a more manageable graph of clusters of public keys representing distinct entities.
How to identify users? [MPGLMVS IMC13]

Users can use arbitrarily many public keys (pseudonyms); as a result the Bitcoin graph is complicated and has 12 million public keys.

Collapse into a more manageable graph of clusters of public keys representing distinct entities.

Collect ground truth data by participating in transactions.
Clustering by inputs
Clustering by inputs

1 7
2 15
3
Clustering by inputs

1  7  15
Clustering by inputs
Clustering by inputs

Heuristic #1: the same user controls these addresses 
[N08, RH11, RS13, A+13]
Change addresses

1  7  
2  
3  15
Change addresses

1
2
3

7
15

Cluster

Diagram
Change addresses
Change addresses
Change addresses

All bitcoins received in a transaction **must be spent all at once**
Change addresses

All bitcoins received in a transaction must be spent all at once
Change addresses

1
2
3
7
15

All bitcoins received in a transaction must be spent all at once
All bitcoins received in a transaction must be spent all at once
Change addresses

All bitcoins received in a transaction must be spent all at once
Change addresses

All bitcoins received in a transaction **must be spent all at once**

In the standard idiom, change addresses are **used at most twice**: to receive change and to spend it
Clustering by change
Clustering by change

identify using one-time behavior

.pk

1

14

14
Clustering by change

Heuristic #2: the same user also controls this address
Data collection
Data collection

Engaged in transactions with:
Engaged in transactions with:

- Exchanges
Data collection

Engaged in transactions with:

- Exchanges
- Vendors
Data collection

Engaged in transactions with:

- Exchanges
- Mining pools
- Vendors
Data collection

Engaged in transactions with:

• Exchanges
  ![MT.GOX](image)
• Vendors
  ![bitpay](image)
• Mining pools
  ![50BTC](image)
• Gambling sites
  ![satoshiDICE](image)
Data collection

Engaged in transactions with:

- Exchanges
- Mining pools
- Wallet services
- Vendors
- Gambling sites
Data collection

Engaged in transactions with:

- Exchanges
- Mining pools
- Wallet services
- Vendors
- Gambling sites
- Mix services
Data collection

Engaged in transactions with:

- Exchanges
  ![MT.GOX logo]

- Mining pools
  ![50BTC logo]

- Wallet services
  ![StrongCoin logo]

- Vendors
  ![bitpay logo]

- Gambling sites
  ![satoshiDICE logo]

- Mix services
  ![BITCOIN FOG logo]

Scraped published tags
Data collection

Engaged in transactions with:

- Exchanges
- Mining pools
- Wallet services
- Vendors
- Gambling sites
- Mix services

Scraped published tags

Found addresses discussed on forums
Exchanges

- Bitcoin-24x
- Bitcoin-Central
- bitcoin.de
- Bitcurex
- bitfloor
- BitMarket.eu
- BITME
- BITSTAMP
- Bitcoin China
- BTC e
- CAMP BX
- VirtEx
- iCBIT
- mercadobitcoin
- MT.GOX
- THE ROCK
- Vircurex
- BitInstant
- BITCOIN NORDIC
- btcQuick
- FastCash4Bitcoins
- LILION TRANSFER
- NANAIMO GOLD
- OKPAY
Vendors
Putting it all together
Putting it all together
Putting it all together
Putting it all together

Transact
us -> them

Cluster
Putting it all together
Putting it all together
Putting it all together

Transact

Cluster

Bootstrap

us \rightarrow \text{them}
Putting it all together

Interacted with 31 MtGox addresses, tagged 518,723!

Participated in 344 transactions and tagged 1.3M public keys
Clustering using Heuristic 2
Clustering using Heuristic 2

bicycle wheel with gambling at center

satoshi dice
btc dice
clone dice

mtgox
instawallet
silk road
Clustering using Heuristic 2

bicycle wheel with gambling at center

strongly connected component with most of our named users
Following bitcoins
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries

change address
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries

Identifying recipients potentially de-anonymizes user
Following bitcoins

Can see when bitcoins meaningfully cross cluster boundaries

Identifying recipients potentially de-anonymizes user

Hypothesis: if you subpoena exchanges, you can identify users
Tracking technique
Tracking technique

tracking heists

= exchange
Tracking technique

tracking heists

\[ \text{HD} \text{M}^+ \text{ NDSS’14} \]

\( \bigcirc = \text{exchange} \)
Tracking technique

tracking heists

[HDMM+ NDSS’14]

○ = exchange

individual thefts

MT.GOX

MT.GOX
Tracking technique

[HDMM+ NDSS’14]

tracking heists

= exchange

individual thefts

service interaction

Follow The Bitcoins: How We Got Busted Buying Drugs On Silk Road's Black Market
Tracking technique

tracking heists

>1,794 BTC

≈2,084 BTC

[HDM+ NDSS’14]

individual thefts

service interaction

exchange

MT.GOX

Follow The Bitcoins: How We Got Busted Buying Drugs On Silk Road's Black Market
Tracking technique

tracking heists

[HDM+ NDSS’14]

>1,794 BTC

≈2,084 BTC

We traced over $3M back to illicit activities!
Takeaway

How much anonymity does Bitcoin really provide?
Takeaway

How much anonymity does Bitcoin really provide?

Our analysis provides a real-world way to track flows of bitcoins
Takeaway

How much anonymity does Bitcoin really provide?

Our analysis provides a real-world way to track flows of bitcoins.

Seems hard to launder significant quantities of money.
My research

[MSF10,LM13]  [BMT14]  [MEKHL10]  [MMCS11]  [MMS11]

[CM14]  [CKLM12,13a,13b]  [CMZ13]  [MP+13,HDM+14]  [OMSK13]

RKA  Bitcoin
Acknowledgements
Acknowledgements

Thanks! Any questions?